CS 297 Report

Yioop! Full Historical Indexing In Cache Navigation

By

Akshat Kukreti

SJSU ID: 008025342

Email: akshat.kukreti@sjsu.edu

Project Advisor: Dr. Chris Pollett

Professor,

Department of Computer Science

San José State University

One Washington Square

San José, CA 95112.

Introduction

The aim of this project is to add new features to the Yioop! search engine that enable users to navigate through Yioop!'s cache of crawled files. When Yioop! displays the results of a query, it also displays a link to the cached version of the web page. One enhancement is to enable the user to access all the cached versions of the page through the given link. This feature is similar to that of The Internet Archive. A second new feature is to modify all the links in cached pages so that they redirect within the index before redirecting to the live page. The third aim of the project is to incorporate Entity Tags (ETags) in Yioop!'s queue server program so that when a re-crawl is done, the ETags can be used to maintain latest version of cached pages.

This report describes the work done during the Fall 2012 semester in preparation for the final project. This consisted of a sequence of deliverables in which I experimented with Yioop!'s caching mechanism and Entity Tags.

For **Deliverable 1**, I modified Yioop!'s cache request and output code so that when a timestamp other than the current timestamp is specified in a URL, it looks for the timestamp nearest to the specified timestamp that has the cached version of the page specified by the URL. For **Deliverable 2**, I built on deliverable 1 by modifying Yioop!'s code for URL canonicalization. The modified code canonicalizes URLs so that they redirect within the index before redirecting to the live site. For **Deliverable 3**, I modified Yioop!'s User Interface for cached versions of web pages. When a cached web page is displayed, links to all cached versions that are present in the archive are also displayed. For **Deliverable 4**, I wrote an experiment using PHP and cURL for extracting ETags from web pages and using them with ETag headers.

Deliverable 1: Modify Cache Request and Output Code so that it looks for a timestamp, nearest in future, to the timestamp specified in the cached request.

Along with search results, Yioop! also displays links to cached web pages. The link to the cached version of a web page also contains a timestamp argument, that corresponds to the time at which the page was cached. Following is an example of a cached link.

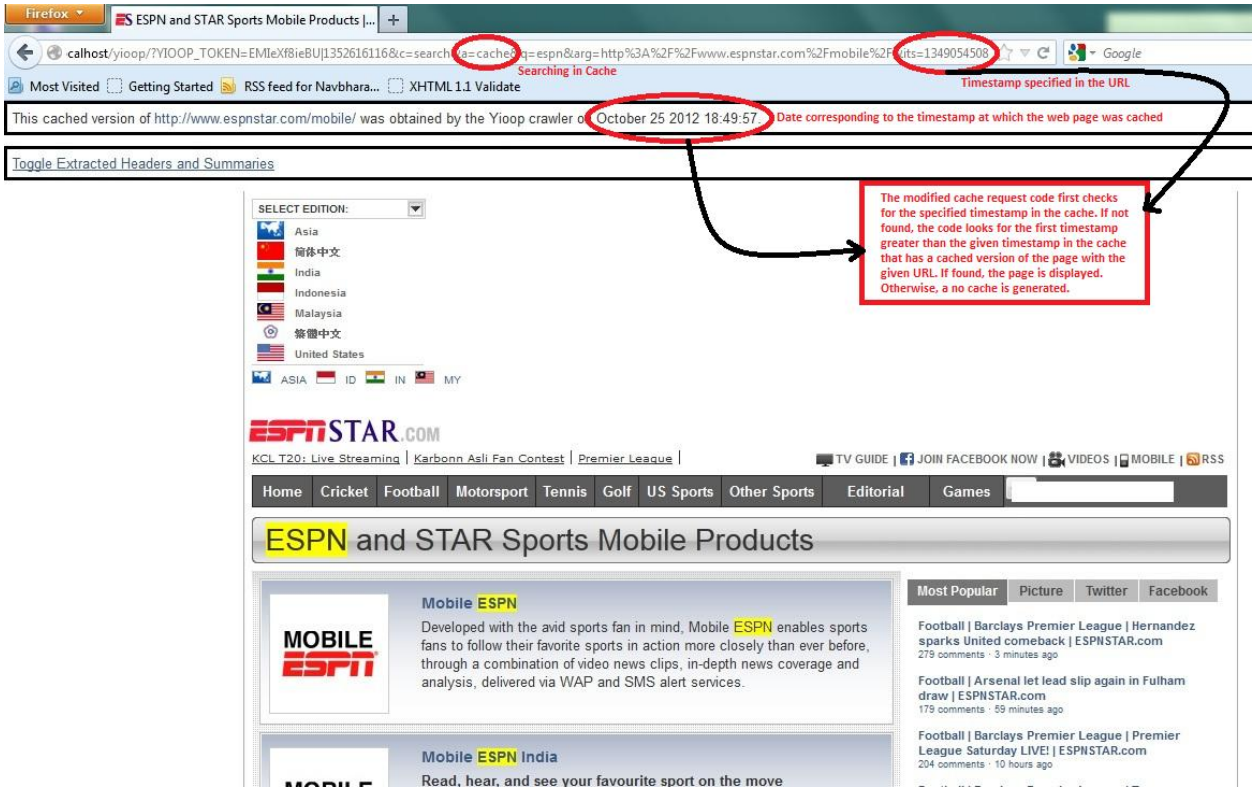
http://www.yioop.com/?YIOOP_TOKEN=7B-qsBHUu34|1354640487&c=search&**a=cache**&q=sjsu&arg=http%3A%2F%2Fwww.sjsu.edu%2F&**its=1336804999**

The parameter a=cache means that the type of search is a search within the cache and the its parameter is the UNIX timestamp corresponding to the cached version of the URL.

If the its parameter is specified, Yioop! looks for the specified timestamp. If a cached version of the page associated with the URL exists for the specified timestamp, it is displayed. Otherwise, a cache not found message is displayed.

For my first deliverable, I modified Yioop!'s code so that first it looks for the specified timestamp. If a cached version is not available for the specified timestamp, a search is made for the nearest future timestamp that has a cached version of the web page. If such a timestamp is found, the associated cached web page is displayed. If such a timestamp is not found, a cache not found message is displayed.

Screenshot for Deliverable 1:



First the timestamp specified in the URL is searched. If not found, the nearest future timestamp that has a cached version of the page is found.

Deliverable 2: Modify links within cached pages so that they redirect within the cache before going to the live site.

In Yioop!, when a link in a cached web page is clicked, it redirects to the live version of the web page. In order to have complete cache navigation, the link should redirect to a page in the cache.

URL canonicalization: Canonicalization is a process by which two quantities can be represented in a standard form. The standard or canonical form can be used to determine whether two or more quantities are similar. In search engines, if a URL is converted into a canonical form, the search engine crawler can make use of the canonical URL to determine if the URL has already been visited. If the crawler finds a URL that has a canonical form similar to a URL that has already been seen, the crawler doesn't visit the URL.

For Deliverable 2, I made changes to the code that is responsible for canonicalization of URLs. I modified the URLs to include the index timestamp of their respective index. I also added an identifier to the URLs. On clicking a URL on a cached page, the specified timestamp is looked up. If the timestamp is not found, the nearest future timestamp is looked up that has a cached version of the web page for the URL. If no such timestamp is found, the identifier tacked to the URL is used to redirect it to the live version of the web page.

Following is an example of a link within a cached page.

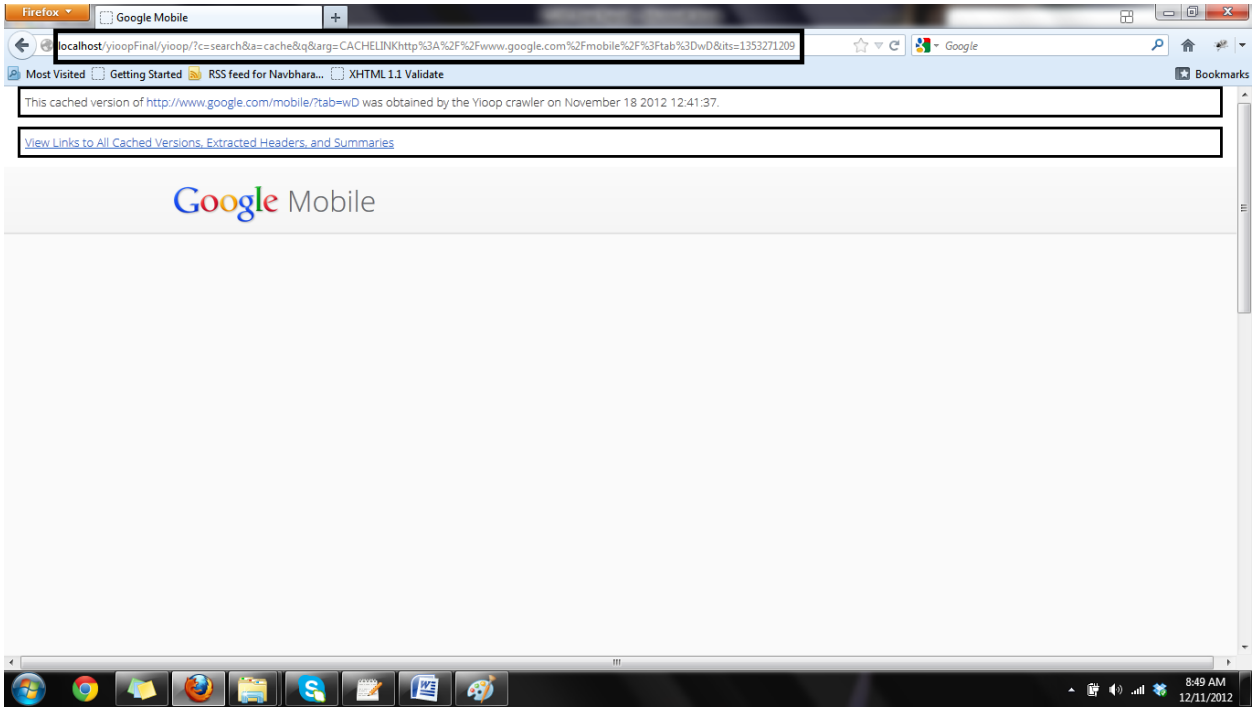
http://localhost/yioopFinal/?c=search&a=cache&q&arg=**CACHELINK***http%3A%2F%2Fwww.google.com%2Fmobile%2F%3Ftab%3DwD*&**its=1353185182**

The its parameter is the same as seen in Deliverable 1. CACHELINK is the identifier that identifies links within a cached page.

Screenshot 1 for Deliverable 2:

The modified link first searches in the cache.

Screenshot 2 for Deliverable 2:



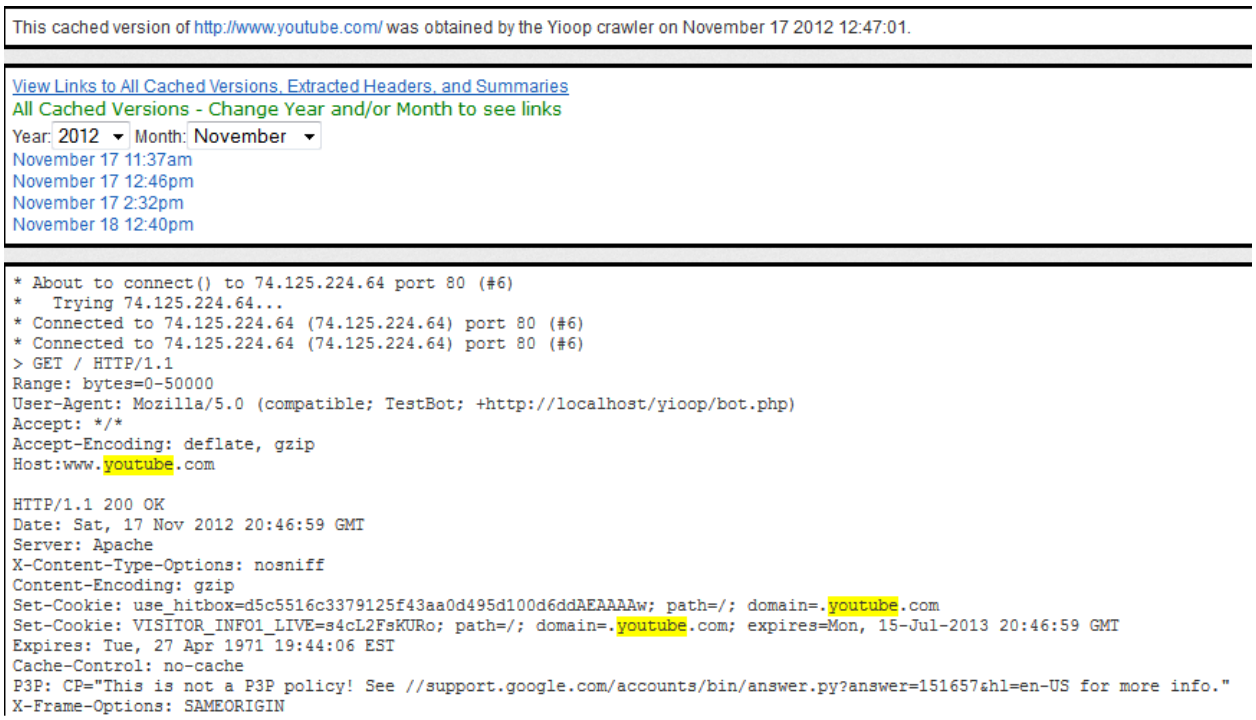
On clicking the link shown on the previous screenshot, the cached version of the page was displayed.

Deliverable 3: Display links to all cached pages on the cached version of a web page.

In Yioop!, when the cached version of a web page is displayed, one way to navigate to another cached page is by changing the timestamp parameter. A more user friendly way to do the same would be through an interface that allows users to select the date for which they wish to see the cached page.

For my third deliverable, I extended Deliverable 1 by extracting links to all cached versions for a given web page. I then created a User Interface for displaying the extracted links. The Interface allows users to choose the year and month for which they wish to see links for the cached page. On choosing a year and month, the links for the specified year and month combination are displayed in the form of a list. This feature is similar to The Internet Archive as it allows users to choose a time in the past, and view the snapshot of their desired webpage taken at that time.

Screenshot for Deliverable 3:



This cached version of http://www.youtube.com/ was obtained by the Yioop crawler on November 17 2012 12:47:01.

View Links to All Cached Versions, Extracted Headers, and Summaries
All Cached Versions - Change Year and/or Month to see links
Year: 2012 Month: November
November 17 11:37am
November 17 12:46pm
November 17 2:32pm
November 18 12:40pm

```
* About to connect() to 74.125.224.64 port 80 (#6)
*   Trying 74.125.224.64...
* Connected to 74.125.224.64 (74.125.224.64) port 80 (#6)
* Connected to 74.125.224.64 (74.125.224.64) port 80 (#6)
> GET / HTTP/1.1
Range: bytes=0-50000
User-Agent: Mozilla/5.0 (compatible; TestBot; +http://localhost/yioop/bot.php)
Accept: */*
Accept-Encoding: deflate, gzip
Host:www.youtube.com

HTTP/1.1 200 OK
Date: Sat, 17 Nov 2012 20:46:59 GMT
Server: Apache
X-Content-Type-Options: nosniff
Content-Encoding: gzip
Set-Cookie: use_hitbox=d5c5516c3379125f43aa0d495d100d6ddAEAAAAw; path=/; domain=.youtube.com
Set-Cookie: VISITOR_INFO1_LIVE=s4cL2FsKURo; path=/; domain=.youtube.com; expires=Mon, 15-Jul-2013 20:46:59 GMT
Expires: Tue, 27 Apr 1971 19:44:06 EST
Cache-Control: no-cache
P3P: CP="This is not a P3P policy! See //support.google.com/accounts/bin/answer.py?answer=151657&hl=en-US for more info."
X-Frame-Options: SAMEORIGIN
```

The screenshot shows that there are four snapshots for November 2012 for http://www.youtube.com/.

Deliverable 4: ETag experiments

Entity Tags: An Entity tag is a unique identifier associated with a resource available on the web. It is a part of the HTTP protocol and can be used to determine if a resource has changed over time. An ETag can be used with the HTTP header when making an HTTP request. Following are the headers available for ETags:

1. If-Match:"ETag value": If "ETag value" matches the ETag of the resource, entire contents of the resource are returned. Otherwise, a **Status 402 (precondition failed)** is returned.

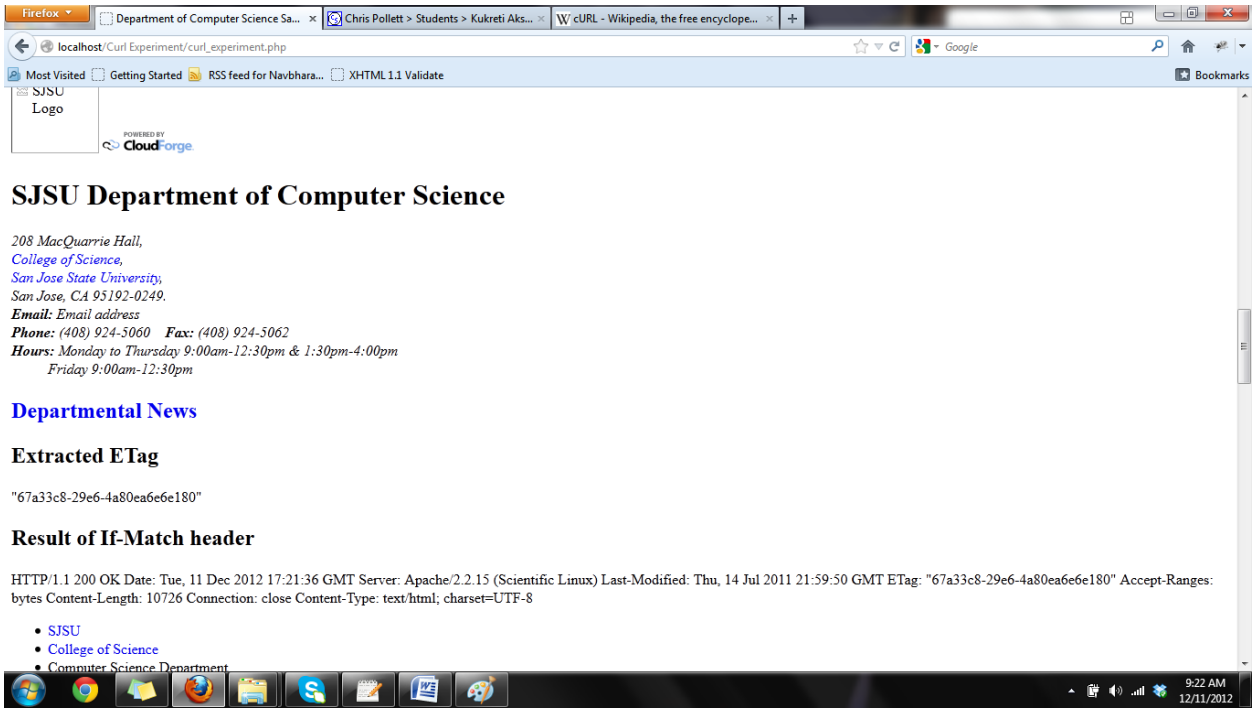
2. If-None-Match:"ETag value": If "ETag value" matches the ETag of the resource or if * is given, it means that the resource hasn't changed. In that case a **Status 304 (not modified)** is returned. Otherwise, entire contents of the resource are returned.

For my last deliverable, I wrote an experiment for extracting ETags from web pages. The code was written using **PHP** and **cURL**(a library for transferring data using various protocols, including HTTP).

The program downloads the contents of a webpage using HTTP. It then checks if the page headers include an ETag. If found, the ETag is extracted. After extracting the ETag, the program attaches the ETag value to the ETag headers discussed above. The headers are then used for making requests for the web page downloaded earlier.

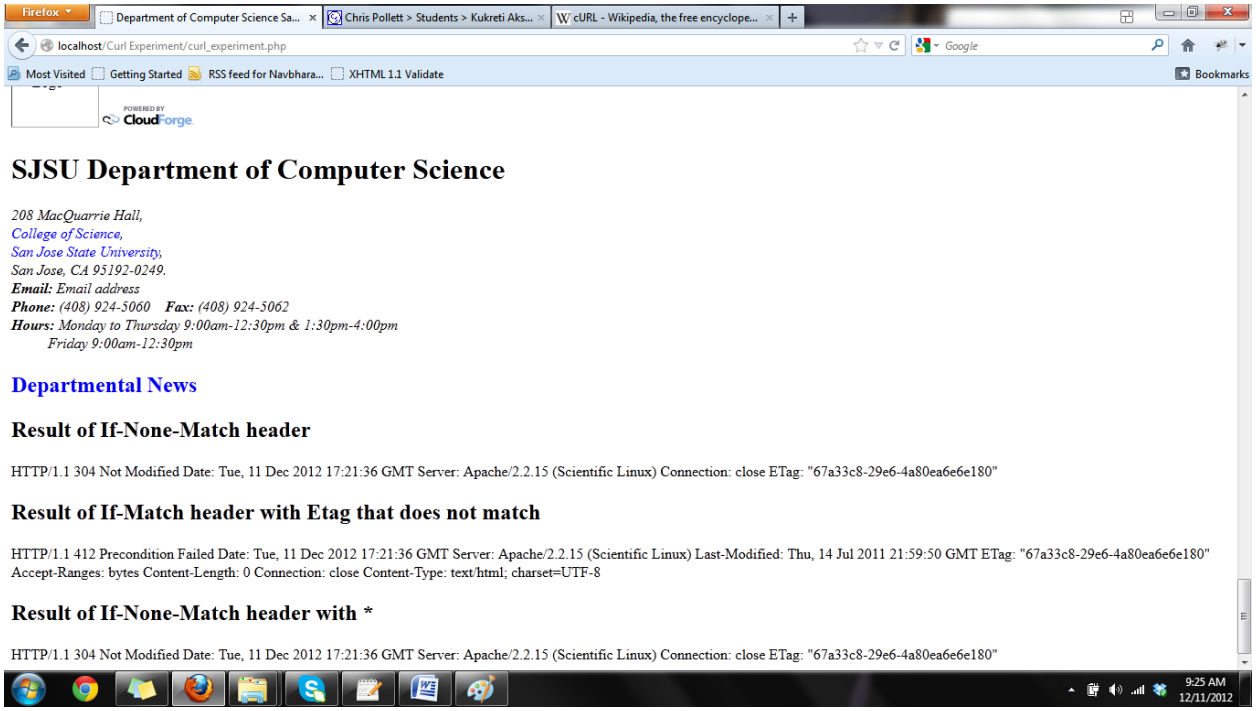
Screenshots for Deliverable 4:

1. Etag extraction



The contents of the web page are downloaded and the ETag is extracted from the header.

2. Results of using the extracted ETag value with ETag headers.



The screenshot shows the responses obtained for different ETag headers.

Conclusion

This semester, I experimented with Yioop!'s cache navigation. For Deliverable 1, I changed how Yioop! handles request for cached pages to enable searching across multiple indexes. For Deliverable 2, I extended Deliverable 1 by modifying links within cached pages so that they search for a cached version of the page before redirecting to the live page. For Deliverable 3, I created a user interface that allows user to view links to all cached versions of a web page. Users can select the desired year and month and view links accordingly. For my last deliverable, I experimented with ETags using PHP and cURL. Through this project I gained an understanding of Yioop!'s mechanism for handling cache requests. I also learned about Entity tags and how they can be used to make sure that a web page is up to date.

Next semester, I will incorporate ETags in Yioop!'s queue server program so that when a re-crawl is done, ETags can be used to make sure that the index is up to date. The re-crawl will be done incrementally based on the number of index partitions. For example, after creating ten index partitions, Partition 1 is re-crawled. After twenty index partitions, Partitions 1 and 2 are re-crawled, and so on.

References

1. [Buttcher 10] Information Retrieval: Implementing and Evaluating Search Engines. Stefan Buttcher, Charles L.A. Clarke, Gordon V. Cormack. The MIT Press. 2010.

2. [Kahle 96] Archiving the Internet. Brewster Kahle. Internet Archive. 1996.

3. [Rackley 09] Internet Archive. Marilyn Rackley. Library, Harvard University, Cambridge, Massachusetts, U.S.A. 2009.